Project Coding Standards

|  |  |  |
| --- | --- | --- |
| Element | Format | Example |
| Class | camelCase | camelCase |
| Interface | camelCase | camelCase |
| Static Object | camelCase | camelCase |
| Method Parameter | camelCase | userName |
| Variable | camelCase | firstName |
| Constant | camelCase | camelCase |
| Namespace | camelCase | Property.Basement |
| Property | camelCase | camelCase |
| Method | camelCase | camelCase |
| Enumeration | camelCase | camelCase |
| Event | camelCase | camelCase |
| Database Table | PascalCase | Order |
| Database Column | under\_line | order\_id |

# Identifier Formatting

## Object Naming

The object names will be prefixed with the abbreviated name of the type of object. The objects themselves will be named with entire words with no abbreviations

Examples:

* TextBox – firstName
* Label – firstName
* Button – firstName

## Variables/Constants

* All variables will be created on separate lines
* All variables will be initialized after they are created, on a separate line
* All variables should be accessed using properties outside of the class, not directly
* All constants will be declared globally (class scope)

Examples:

* var featured;

featured=””;

## Curly Braces

* All of the code between curly braces should be indented 1 tab space (including nested brackets)

## Spacing

* A blank space should appear after each comma in a comma-separated list
* A line of code should be split into several lines if it extends further than the code window
* A space should appear before and after operators such as > < =.

## Methods

* All method parameters should be of a certain type, not generic like object
* Any methods that return a value should only have one return statement
* A single method should fulfill a single task and should not be broad in its purpose
* Any major operations (i.e. DB Access) should not be handled within an event handler, and should be called in a separate method
* Documentation comments should appear above each method indicating its purpose, parameters and return value
* Implementation comments will be provided for explaining complex logic

## Exception Handling

* A try catch block should be looking for a specific exception, not a general exception
* The catch block should not be left empty
* A try-catch block should enclose the entire method code, only the part that will throw an exception

## Others

* Use the reformat code feature in webstorm.